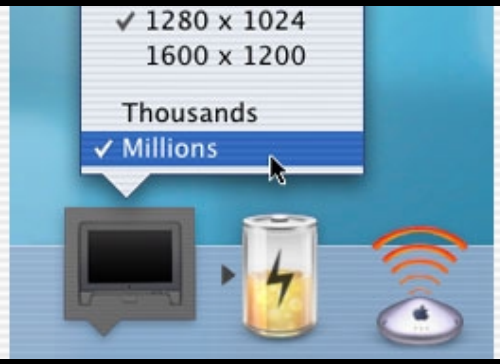
Session 110

Drawing Unicode Text With ATSUI



John H. Jenkins
Senior Software Engineer

What We'll Cover

- What ATSUI is and what it does
- Who is using it now
- When to use it
- ATSUI basics
- ATSUI examples

What Is ATSUI?

- Apple Type Services for Unicode™ Imaging
- Text drawing APIs available directly or indirectly
  - To access Quartz text effects on Carbon
  - To draw Unicode text
  - To provide high-end typography
  - Ask for one, get the others at no extra cost
- The right way to display text on Mac OS X

ATSUI and Quartz

- Fully integrated
- Respects settings in CGContext
  - Transparency
  - Skew, rotation (matrix control)…
- Uses Quartz text rendering attributes
- Quartz anti-aliasing

ATSUI and Unicode

- Supports all of Unicode 3.0
  - New scripts, new characters (beyond those available in WorldScript)
- Automatically handles combining characters
  - Accents for Latin, Greek, Cyrillic
- Automatically handles complex scripts
  - Middle East and South Asia

ATSUI and Unicode

- No need to rewrite application for international text
- Scripts you get for free:
  - Latin (much larger set than Latin—1 or MacRoman), CJK Ideographs, Kana, Hangul, Greek, Cyrillic, Armenian, Georgian, Ethiopic, Cherokee, Inuktitut, Ogham, Runic, Yi, IPA, and many others…

ATSUI and Typography

- Takes advantage of data in the font
  - Default settings can be overridden
- Ligatures
  - More than just fi and fl
- Swashes and alternate forms
  - Letters can change their shape
  - Shape changes can be context-sensitive

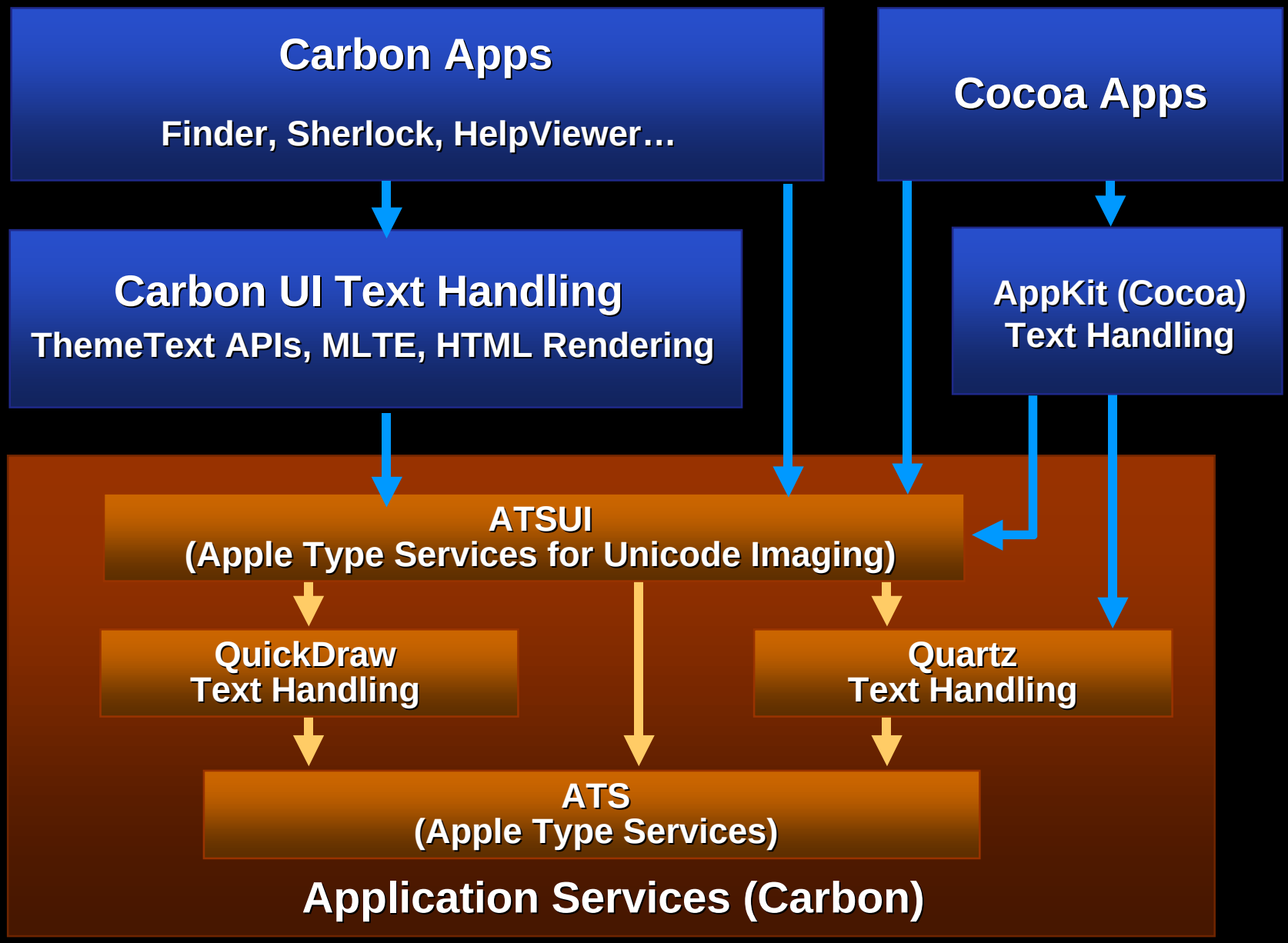
ATSUI and Typography

- Advanced features formerly limited to advanced desktop publishing now available to everybody
  - Optical alignment
  - Hanging punctuation
  - Kerning and tracking
  - Sophisticated justification

code
日本語

Zapfino

Beautiful Type


Carbon Apps
Finder, Sherlock, HelpViewer…
Cocoa Apps
Carbon UI Text Handling
ThemeText APIs, MLTE, HTML Rendering
AppKit (Cocoa)
Text Handling
ATSUI
(Apple Type Services for Unicode Imaging)
QuickDraw
Text Handling
Quartz
Text Handling
ATS
(Apple Type Services)
Application Services (Carbon)

Who Uses ATSUI?

- Finder (editing file names)

- Toolbox (all HI text for Carbon and Cocoa)
- Cocoa (calls into ATSUI for Unicode layout)
- Help Viewer (help text)
- Sherlock (search results)
- Multi-Lingual Text Engine (MLTE)
- Many more…
- Unicode is Mac OS X's native character set

When Do I Use ATSUI Directly?

- Need fine control
  - Glyph positions, text at an angle or on a path,line positioning
- Need certain ATSUI features
  - Vertical, justification control, optical alignment…
- Writing your own text editing engine
  - Can’t use MLTE but need Unicode or ATSUI features

ATSUI Features

- Supports everything in Unicode 3.0 (as long as you have appropriate fonts)
- Supports vertical text (CJK)
- Supports advanced typography
  - Kerning, optical alignment, variation fonts, ligatures, glyph alternatives, baseline adjust…
- APIs to help you support editing
  - Highlighting, hit mapping, cursor movement
- Low-level access to glyph outlines

How Does ATSUI Work?

- Unicode text (UTF-16)
  - Use TEC if necessary to convert
  - You own the buffer
- ATSUStyle
- ATSUTextLayout

ATSUStyle…

- Opaque structure containing:
  - ATSUFontID
  - Size, color
  - Font features
  - …
- An ATSUStyle is reusable
- Recommended that you keep these around and share them if possible—they are lightweight

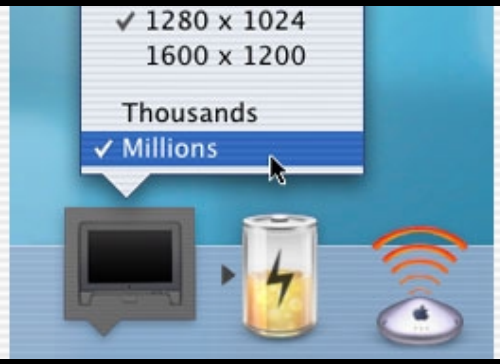
ATSUTextLayout…

- Opaque structure which ties together text and runs of ATSUStyles
- One or more paragraphs (for bi-directional processing)
- Adds line attributes, rotation, etc.
- Should cache if possible

ATSUI Controls (Examples)

- Style Attributes
  - Font, size, color, vertical, with- and cross-stream shift, kern control, optical, hanging…
  - Font features (defined by font)
    - Ligatures, swashes, variant glyphs…
  - Font Variations (defined by font)
    - Continuously variable weight, width…
- Line/Layout controls
  - Rotation, justification, flush, baseline…

ATSUI: How to…



Peter K. Edberg
Senior Software Engineer

Case Study 1: CG and ATSUI

- Create style(s), layout objects
- Add attributes to styles (bulk setter)
  - Here: font, size
- Apply styles to runs in layout
- Add controls to layout (bulk setter)
  - Here: kATSUCGContextTag
- Draw

ATSUI Features Used

- kATSUFontTag
- kATSUSizeTag
- kATSUCGContextTag

C Code—Creating a Style

```
ATSUStyle    style1;

ATSUCreateStyle( &style1 );
```

C Code—Set Style Attributes

- Step 1, set up data for attributes

```
ATSUFontID  fontID;
Fixed       fontSize;
ATSUAttributeTag        iTags[ ];
ByteCount               iSizes[ ]
ATSUAttributeValuePtr   iValuePtrs[ ];
ItemCount   tagCount;

fontID = ...
fontSize = ...
```

C Code—Set Style Attributes

- Step 2, pass data to ATSUSetAttributes

```
iTags[0] = kATSUFontTag;
iSizes[0] = sizeof(ATSUFontID);
iValuePtrs[0] = &fontID;
iTags[1] = kATSUSizeTag;
iSizes[1] = sizeof(Fixed);
iValuePtrs[1] = &fontSize;
tagCount = 2;

ATSUSetAttributes(
        style1, tagCount, iTags, iSizes, iValuePtrs);
```

C Code—Creating a Layout

- One or more style runs (bulk setter)
- You own the text; it isn't copied

```
ATSUTextLayout layout;
ItemCount styleRunCount;

styleRunCount = 1;
ATSUCreateTextLayoutWithTextPtr(
        (UniChar *)string, offset, length, totalLength,
         styleRunCount, &length, &style1, &layout );
```

C Code—Set Layout Control

```
iTags[0] = kATSUCGContextTag;
iSizes[0] = sizeof(CGContextRef);
iValuePtrs[0] = &myCGContext;
tagCount = 1;

ATSUSetLayoutControls( mLayoutBasic,
        tagCount, iTags, iSizes, iValuePtrs );
```

C Code—Draw with CG Effects

- Before drawing, set up CG effects (e.g., rotation)
- Draw multiple items if desired
- Flush the CGContext to make text appear

```
CGContextRotateCTM( myCGContext, angle );
ATSUDrawText( layout,
        kATSUFromTextBeginning, kATSUToTextEnd,
        fixedX, fixedY );
CGContextFlush( myCGContext );
```

On-screen Result

Case Study 2: Equation Display

- Create multiple style objects
  - One for normal text (e.g., normStyle)
  - One for subscripts, if used (e.g., subStyle)
    - Smaller font size, shifted down
  - One for superscripts, if used (e.g., superStyle)
    - Smaller font size, shifted up

ATSUI Features Used

- kATSUCrossStreamShiftTag
- kATSUFontSizeTag

C Code—Styles

```
ATSUStyle normStyle, subStyle;

ATSUCreateStyle( &normStyle );
iTags[0] = kATSUFontTag; …
iTags[1] = kATSUSizeTag;  …
ATSUSetAttributes( normStyle, 2,
        iTags, iSizes, iValuePtrs );

ATSUCreateAndCopyStyle( normStyle, &subStyle );
iTags[0] = kATSUCrossStreamShiftTag; …
iTags[1] = kATSUSizeTag; …
ATSUSetAttributes( subStyle, 2,
        iTags, iSizes, iValuePtrs );
```

C Code—Layout

- Create layout with text and normStyle
  - All text is normal to start with

```
ATSUCreateTextLayoutWithTextPtr(
    (UniChar *)string, offset, length, totalLength,
    1, &length, &normStyle, &layout );
```

- Set subStyle for every run of subscript text

```
ATSUSetRunStyle( layout, subStyle, begin, end );
```

- Draw

On-screen Result

$$G_{\mu\nu} = kT_{\mu\nu}$$

Case Study 3: Vertical Text

- Create layout and style objects as usual
- Set the style to use vertical forms (usually rotated glyphs)
- Then rotate the line using either CG or ATSUI line rotation

ATSUI Features Used

- kATSUVerticalCharacterTag
- kATSULineRotationTag

C Code—Style and Layout

```
ATSUVerticalCharacterType verticalType;

verticalType = kATSUStronglyVertical;
iTags[0] = kATSUVerticalCharacterTag;
iSizes[0] = sizeof(ATSUVerticalCharacterType);
iValuePtrs[0] = &verticalType;

ATSUSetAttributes( vertStyle, 1,
        iTags, iSizes , iValuePtrs );
ATSUCreateTextLayoutWithTextPtr(
       (UniChar *)string, offset, length, totalLength,
       1, &length, &vertStyle, &layout );
```

C Code—Rotation

- For ATSUI rotation (to use ATSUI editing APIs)

```
Fixed angle = FloatToFixed(-90.0);   // degrees
iTags[0] = kATSULineRotationTag;
iSizes[0] = sizeof(Fixed);
iValuePtrs[0] = &angle;
ATSUSetLayoutControls( layout, 1,
        iTags, iSizes, iValuePtrs );
```

On-screen Result

☆ Ｇ７プラスソ連(日本経済新聞 平成３年７月１９日 金曜日朝刊第一面より)

重い出発 上『西側への統合、半開きの扉 対話を継続、改革見守る』

ゴルバチョフ・ソ連大統領が初めて参加した先進国首脳会議（ロンドン・サミット）は、西側七カ国（Ｇ７）の対ソ支援策の決定と戦略兵器削減条約（ＳＴＡＲＴ）調印で合意という成果をあげて終了した。これまで西側世界をリードしてきたＧ７にソ連を加えた「Ｇ８」の誕生は、冷戦終結後の新しい東西協調体制への出発点となるのか。世界は政治・経済両面にわたる大きな転換期を迎えた。

『冷戦は終わった』

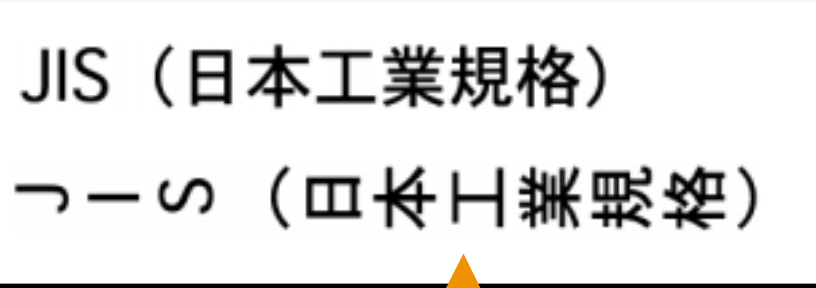
ロンドン・サミットは最終日の十七日、ようやく、本番に入った。主役躍り出たゴルバチョフ大統領は、ブッシュ米大統領との間でモスクワでの首脳会談開催で合意、西側七カ国首脳から対ソ支援の具体策を取りつけた。ブッシュ大統領は「きょうは東西関係にとって歴史的な日となった」と総括した。

ロンドンで開いたサミット、米ソ、Ｇ７プラスソ連の三つの首脳会談を貫いていたのは、「ソ連をどうやって西側世界へ迎えいれるか」というテーマだった。西側への統合の条件は、ソ連が「親しみやすい隣人」となること。ソ連が西側の脅威の対象とならず、すんなりと世界経済に溶け込めるか=西側とゴルバチョフ大統領の双方の対応にかかっていた。

ゴルバチョフ大統領のロンドンへの道のりは長かった。一昨年のアリュシュ・サミットに「参加希望」の書簡を送った同大統領は、ソ連の姿勢変化をかぎとったブッシュ大統領の呼びかけに応じて、年末に地中海・マルタ島での米ソ首脳会談に臨んだ。ゴルバチョフ大統領がそこで「冷戦は終わった」と宣言したのは、いまとなっては西側世界に救いの手を求める最初の叫びだったのかもしれない。

Vertical Forms and Line Rotation

JIS（日本工業規格）

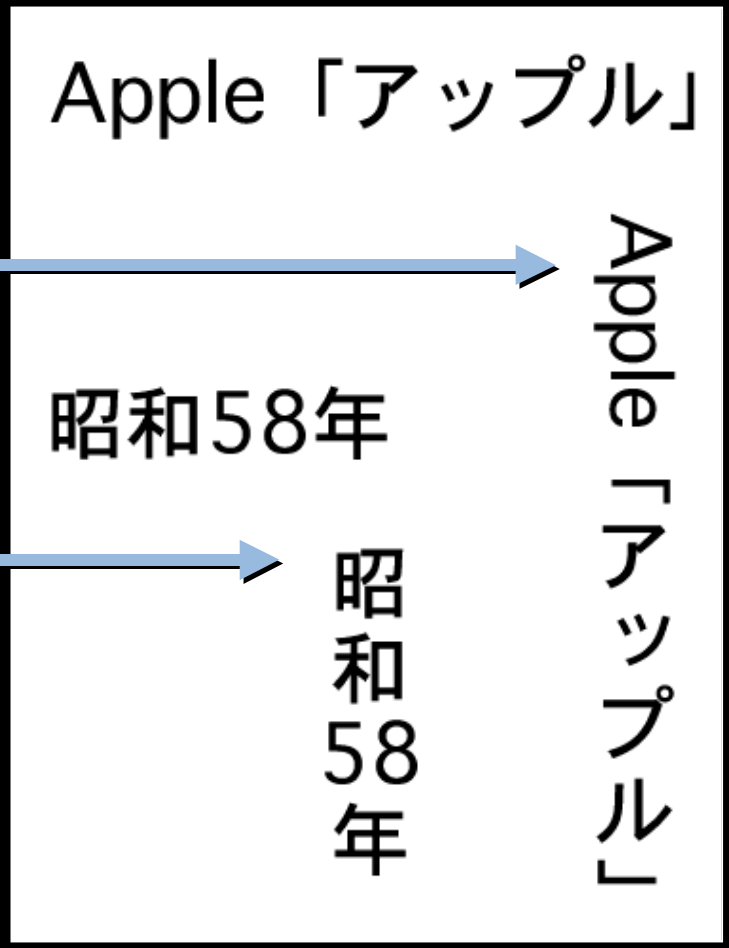


- Vertical forms only (style)
- Line rotation only (layout)
- Vertical forms and line rotation

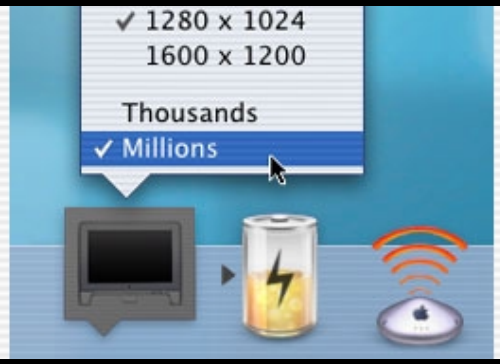
More Sophistication…

- Only rotate some glyphs (e.g. katakana) in a rotated line
- Rotate sections of the line e.g., for Tatechuuyoko

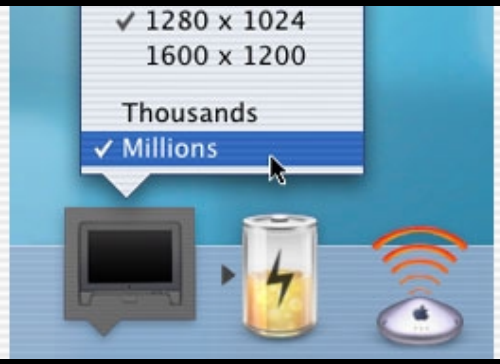
縦中横



Demo



Summary



Xavier Legros
Text & International Technology Manager
Apple Worldwide Developer Relations

Summary

- ATSUI is the heart of text drawing on Mac OS X
- Use it directly when
  - DrawThemeTextBox is not enough
  - You need fine control of the text
  - You need smooth integration with Quartz
  - You're writing a word processor

To Find Out More

- ATSUI Documentation

  http://developer.apple.com/techpubs/macosx/Carbon/text/ATSUI/atsui.html

  http://developer.apple.com/qa/qa2001/qa1027.html

- ATSUI Sample Code

  http://developer.apple.com/samplecode/Sample_Code/Text/TypeServicesForUnicode.htm

Roadmap

113 Text on Mac OS X: The other options (MLTE, Appearance…)	Hall 2 **Wed., 3:30pm**
128 Font Management on Mac OS X: Supporting many technologies and formats	Room C **Thurs., 2:00pm**
FF105 Feedback Forum: **International Technologies:** Are we giving you what you need?	Room C **Thurs., 5:00pm**

Who to Contact

Xavier Legros
Apple Worldwide Developer Relations
xavier@apple.com

Q&A

http://developer.apple.com/events/urls.html

WWDC

Worldwide Developers Conference 2001